Operating Systems Laboratory

Lab 2

Om Patil (200010036)

Hrishikesh Pable (200010037)

Part 2

We studied the behaviour of the Minix scheduler with the use of the Unix Benchmark Suite. Modifying the given `workload_mix,` we created a few custom workload mixes to study the scheduler's behaviour. Each workload contains a different set of benchmarks helping us understand how the scheduler works.

workload_mix1.sh

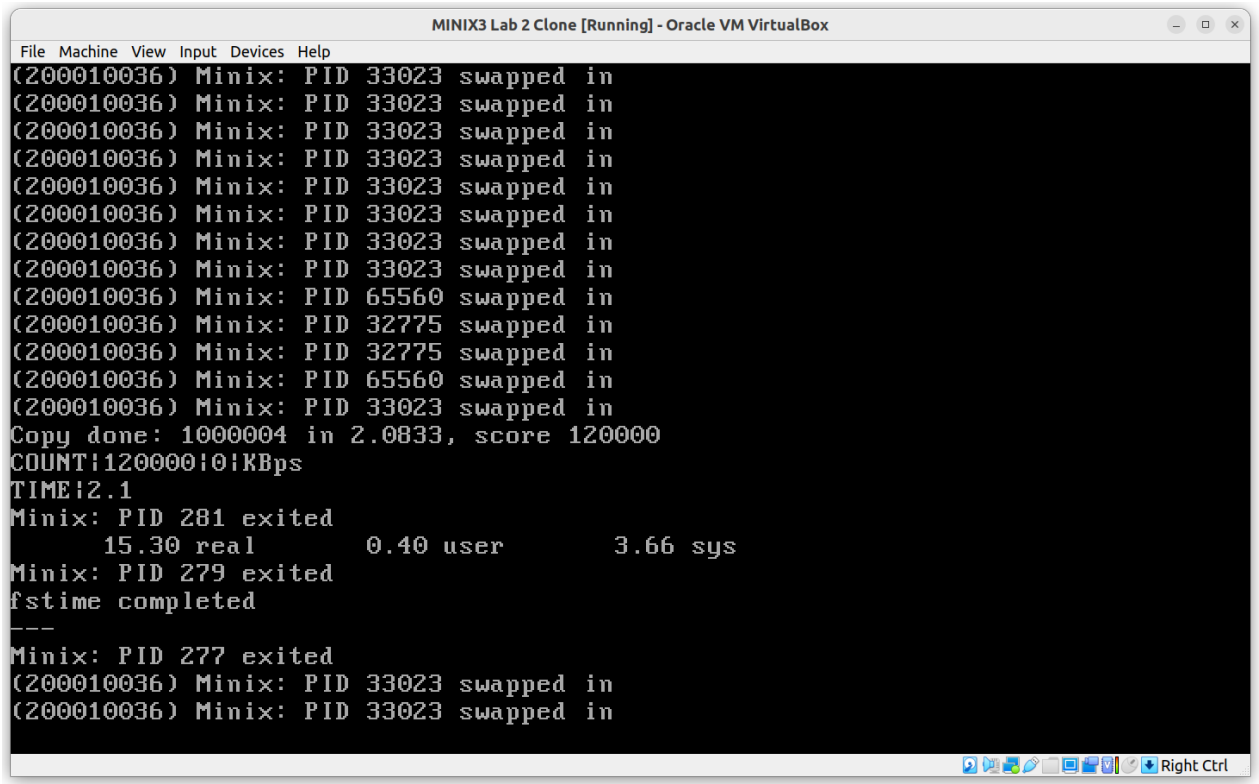
The workload is as follows,

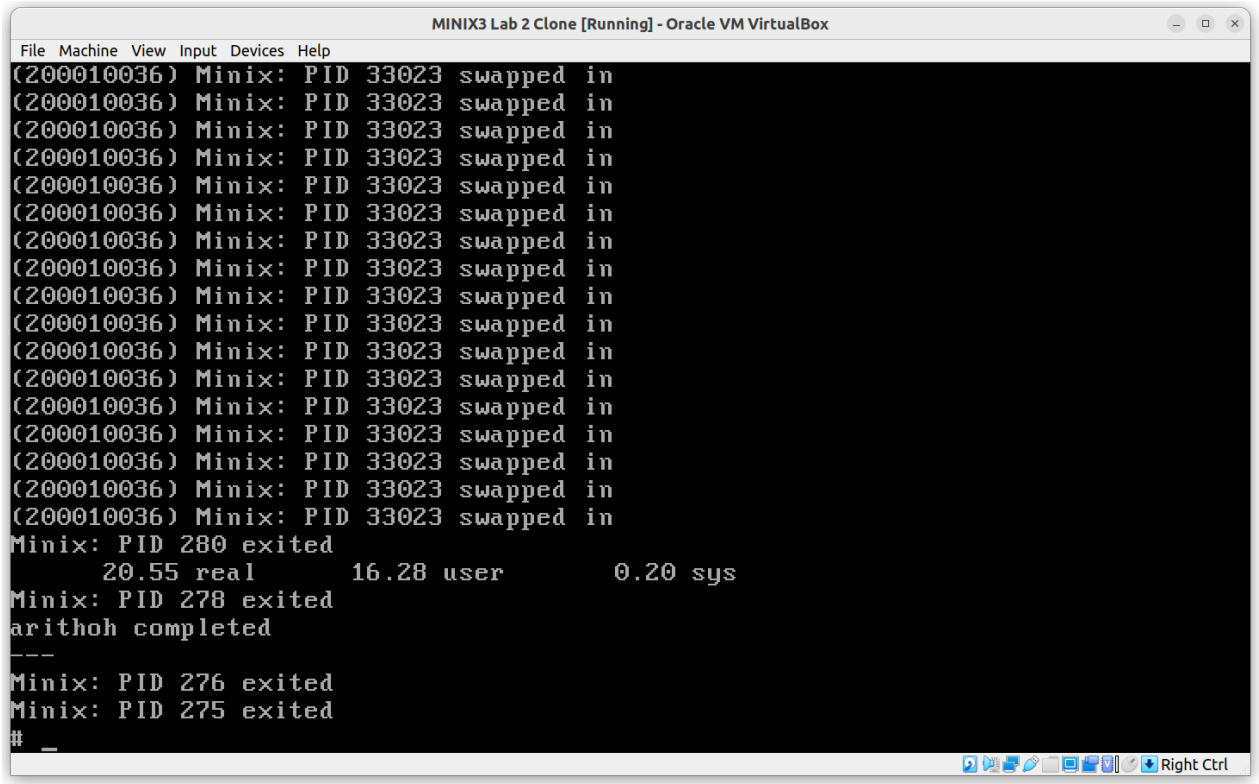
```
#!/bin/sh
./arithoh.sh &
./fstime.sh &
wait
```

The observations are as follows,

`arith.c` involves arithmetic operations making it CPU bound, whereas `fstime.c` involves read and write operations making it IO bound. Hence, `arith.c` frequently gets time slices, since it does not wait for any I/O event and does not get blocked. `fstime.c` needs to wait for I/O events for making read and write operations, and hence rarely gets time slice. Once the I/O event occurs, `fstime.c` exits, and only `arith.c` is scheduled henceforth, which exits once all the arithmetic operations are completed.

This can be observed in the following screenshots,





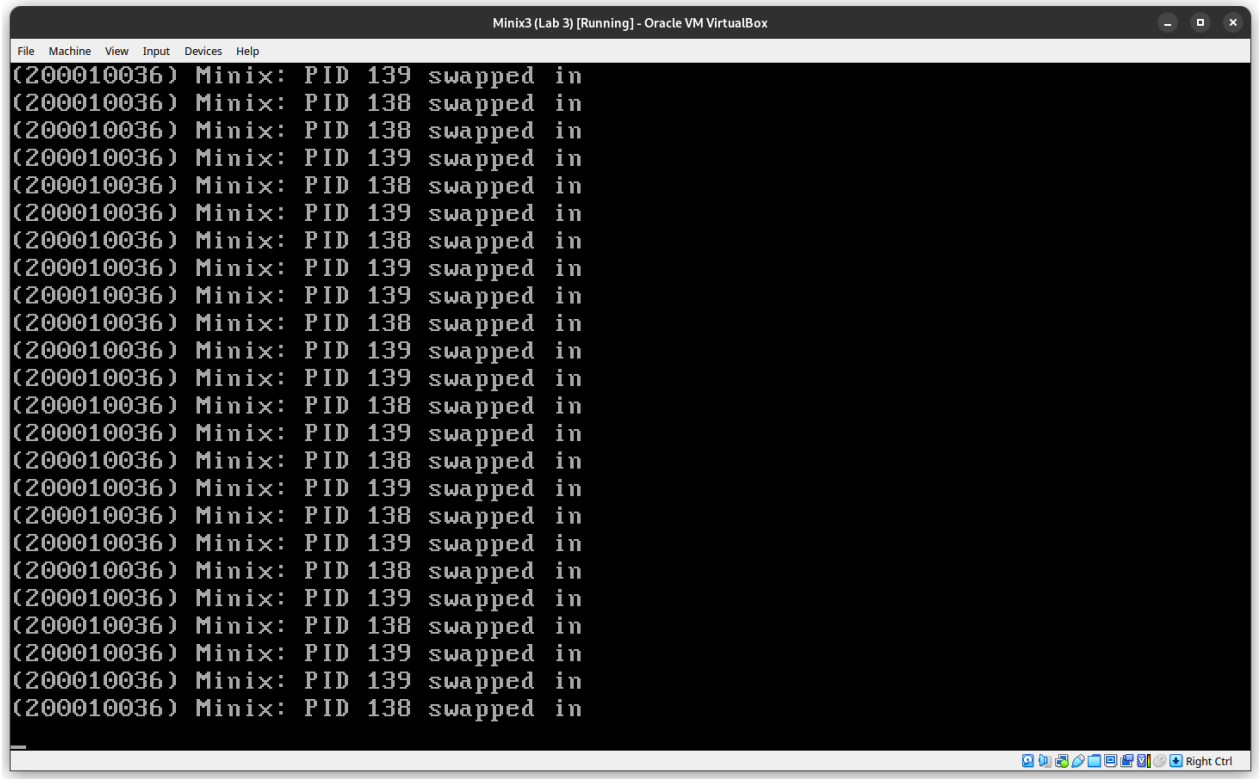
Here 65560 is the PID for the `fstime` process and 33023 is the PID for the `arithoh`.

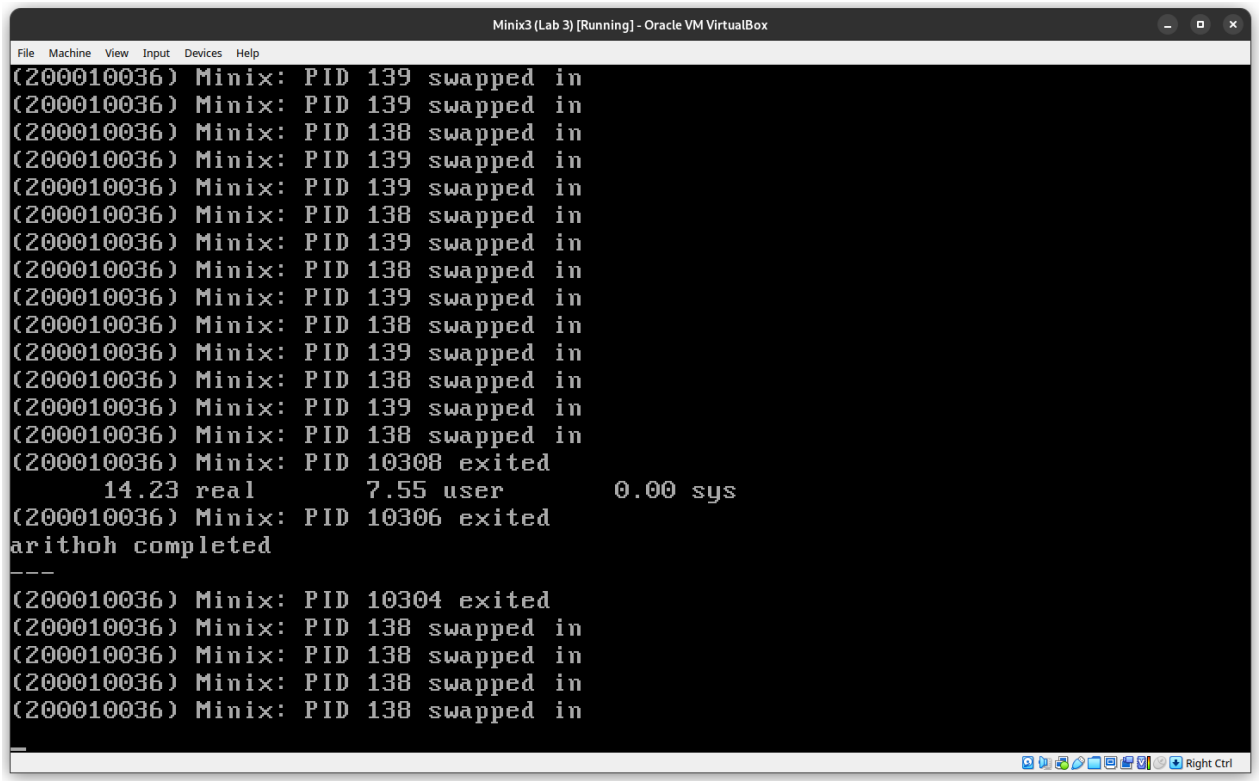
workload_mix2.sh:

The workload is as follows:

```
#!/bin/sh
./arithoh.sh &
./arithoh.sh &
wait
```

In this workload, we run two arith.c files simultaneously. The arith.c processes are CPU-bound processes, which have equal priority, and hence the two processes are allotted alternate time slots by the scheduler, which can be seen in the first screenshot below. Once one of the arith.c processes completes execution, the second process gets the complete CPU time until its completion, as shown in the second screenshot.



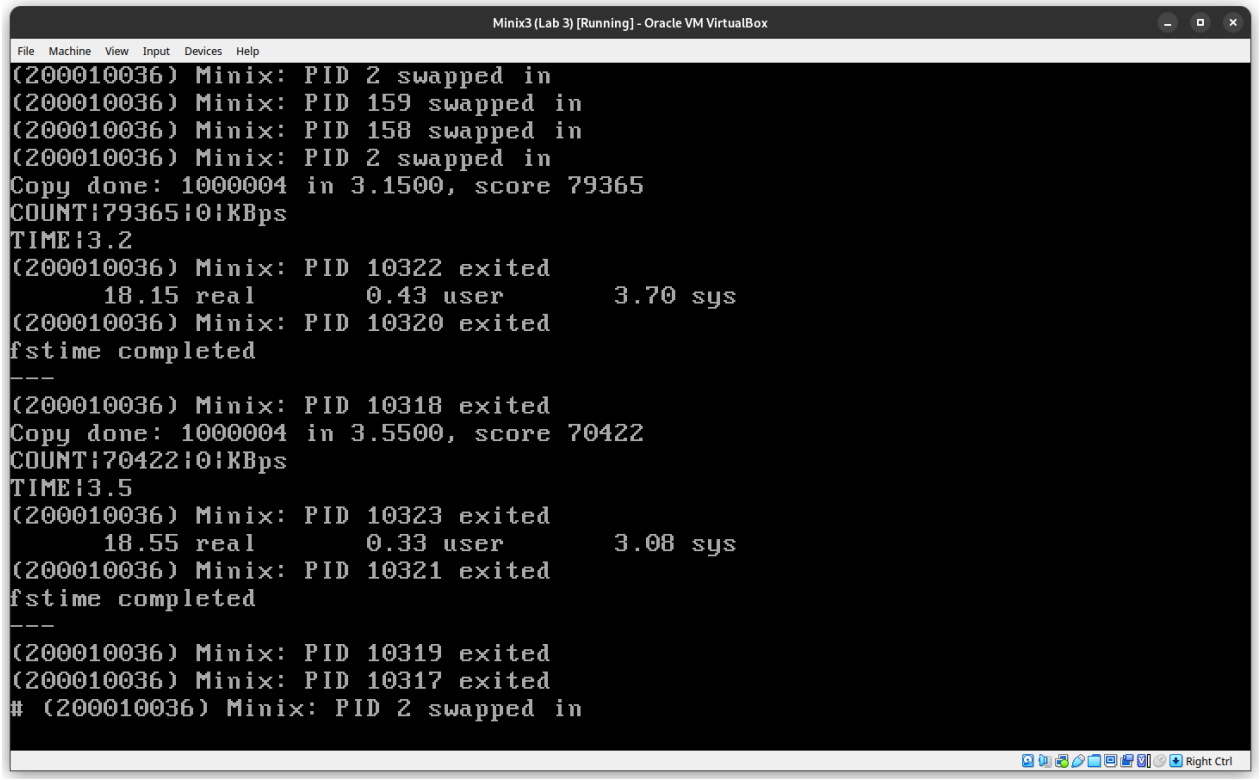


workload_mix3.sh:

The workload is as follows:

```
#!/bin/sh
./fstime.sh &
./fstime.sh &
wait
```

In this workload, we run two `fstime.c` processes simultaneously, which are both I/O bound processes. Both these processes wait for the I/O event. The first process exits as soon as its I/O event is completed. Soon after that, the second process too exits when its I/O event is completed. This can be seen in the following screenshot.


Minix3 (Lab 3) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
(200010036) Minix: PID 2 swapped in
(200010036) Minix: PID 159 swapped in
(200010036) Minix: PID 158 swapped in
(200010036) Minix: PID 2 swapped in
Copy done: 1000004 in 3.1500, score 79365
COUNT¦79365¦0¦KBps
TIME¦3.2
(200010036) Minix: PID 10322 exited
18.15 real 0.43 user 3.70 sys
(200010036) Minix: PID 10320 exited
fstime completed

(200010036) Minix: PID 10318 exited
Copy done: 1000004 in 3.5500, score 70422
COUNT¦70422¦0¦KBps
TIME¦3.5
(200010036) Minix: PID 10323 exited
18.55 real 0.33 user 3.08 sys
(200010036) Minix: PID 10321 exited
fstime completed

(200010036) Minix: PID 10319 exited
(200010036) Minix: PID 10317 exited
(200010036) Minix: PID 2 swapped in
Right Ctrl